Cours de XML - Définition de Type de Document

par G. Chagnon

Date de publication : 20 mars 2009

Dernière mise à jour :

Un fichier XML doit non seulement respecter des règles d'écriture ; il peut aussi, si on le désire, suivre des règles strictes d'enchâssements des éléments. Il existe deux grands langages de description pour ce faire : Les déclarations de type de document (DTD) et les schemas XML

Ce tutoriel s'interresse aux DTD. Une DTD permet de décrire les éléments et leurs attributs autorisés dans un document XML.

I - Introduction

Il peut être parfois nécessaire de préciser les balises et attributs auxquels on a droit lors de la rédaction d'un document **XML**, par exemple si l'on veut pouvoir partager le même type de document avec une communauté d'autres rédacteurs. Deux solutions sont possibles : les « **Schémas XML** » et les « *Document Type Definition* ». Ces dernières sont les plus simples à manipuler et sont apparues en premier, alors que les Schémas n'étaient pas encore définis. Ce sont les raisons pour lesquelles nous allons nous limiter à elles pour le moment. Il faut néanmoins garder à l'esprit qu'il existe une autre solution, plus complexe certes, mais aussi plus puissante. Elle permet notamment d'informer plus efficacement l'utilisateur sur les balises auxquelles il a droit, ou bien de spécifier de manière plus détaillée le format autorisé pour le contenu de l'élément ou de l'attribut.

II - Types de DTD

II-A - Introduction

Une **DTD** peut être stockée dans deux endroits différents. Elle peut être incorporée au document **XML** (elle est alors dite interne), ou bien être un fichier à part (on parle alors de **DTD** *externe*). Cette dernière possibilité permet de la partager entre plusieurs documents **XML**. Il est possible de mêler **DTD** interne et externe.

Il existe de surcroît deux types de **DTD** externes : privé ou public. Les **DTD** privées sont accessibles uniquement en local (sur la machine de développement), tandis que les publiques sont disponibles pour tout le monde, étant accessibles grâce à un URI (*Uniform Resource Identifier*).

Une déclaration de type de document est de la forme :

```
<!DOCTYPE elt.racine ... "..." "...">
```

Nous verrons progressivement par quoi remplacer les points de suspension. Cette déclaration se place juste après le prologue du document. L'élément racine du document **XML** rattaché à cette **DTD** est alors obligatoirement **elt.racine**. Par exemple...

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE commande ... "..." "boncommande.dtd">
<commande>
<item>(...)</item>
<item>(...)</item>
<item>(...)</item>
</commande>
```

II-B - Syntaxe

Le contenu ne change pas suivant le type de **DTD**, mais les déclarations d'une **DTD** interne sont écrites à l'intérieur du prologue du document **XML** alors que celles d'une **DTD** externe sont stockées dans un fichier... externe.

Exemple de déclarations pour une **DTD** interne :

```
<!DOCTYPE biblio[
<!ELEMENT biblio (livre)*>
<!ELEMENT livre (titre, auteur, nb_pages)>
  <!ATTLIST livre
    type (roman | nouvelles | poemes | théâtre) #IMPLIED
    lang CDATA "fr"
  >
<!ELEMENT titre (#PCDATA)>
<!ELEMENT auteur (#PCDATA)>
<!ELEMENT nb_pages (#PCDATA)>
```

II-C - DTD externe

Les deux types de **DTD** externes sont les **DTD** de type public et les **DTD** de type system. Le mot-clef **SYSTEM** indique que le fichier spécifié se trouve sur l'ordinateur local et qu'il est disponible uniquement à titre privé. Le mot-clé **PUBLIC** indique une ressource disponible pour tous sur un serveur distant.

Exemple de déclaration de **DTD** externe de type **SYSTEM** :

```
<!DOCTYPE biblio SYSTEM "bibliographie.dtd">
```

Le fichier associé est le suivant :

```
<!ELEMENT biblio (livre*)>
<!ELEMENT livre (titre, auteur, nb_pages)>
  <!ATTLIST livre
    type (roman | nouvelles | poemes | théâtre) #IMPLIED
    lang CDATA "fr"
  >
<!ELEMENT titre (#PCDATA)>
<!ELEMENT auteur (#PCDATA)>
<!ELEMENT nb_pages (#PCDATA)>
```

Exemple de déclaration de **DTD** externe de type **PUBLIC** :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
```

Dans l'exemple précédent, la chaîne de caractères après le mot PUBLIC fait référence tout d'abord à l'identifiant de la DTD (ici -, qui signifie que la DTD n'a pas de numéro d'enregistrement officiel), au propriétaire de la DTD (ici le W3C), puis son nom, enfin sa langue.

III - Déclarations d'éléments

III-A - Généralités

Une déclaration d'élément est de la forme :

```
<!ELEMENT nom type_element>
```

nom est le nom de l'élément et **type_element** est le type auquel il est associé. Un élément peut être de type texte, vide, séquence ou choix d'élément. Dans ces deux derniers cas, on indique la liste des éléments-enfants. Passons ces types en revue.

III-B - Élément texte

Cet élément est le plus répandu, puisque c'est celui qui contient... du texte. Il se déclare ainsi :

```
<!ELEMENT elt (#PCDATA)>
```

III-C - Élément vide

Un élément vide est, comme son nom l'indique, un élément qui n'a aucun contenu -que ce soit de type texte, ou bien un autre élément. Le mot-clef utilisé pour la déclaration de ce type d'élément est **EMPTY** :

```
<!ELEMENT elt EMPTY>
```

Exemple d'utilisation :

```
<elt />
```

Un élément vide peut cependant fort bien posséder un ou plusieurs attributs. Par exemple

```
<img width="100" height="30" src="paysage.jpg" alt="Paysage de neige" />
```

III-D - Séquence d'éléments

Une séquence d'éléments est une liste ordonnée des éléments qui doivent apparaître en tant qu'éléments-enfants de l'élément que l'on est en train de définir. Ce dernier ne pourra contenir *aucun* autre élément que ceux figurant dans la séquence. Cette liste est composée d'éléments séparés par des virgules et est placée entre parenthèses.

Chaque élément-enfant doit de plus être déclaré par ailleurs dans la **DTD** (avant ou après la définition de la liste, peu importe). Dans le fichier **XML**, ils doivent apparaître dans l'*ordre* de la séquence.

```
<!ELEMENT elt0 (elt1, elt2, elt3)>
```

Exemple d'utilisation valide :

```
<elt0>
  <elt1>(...)</elt1>
```

```
  <elt2>(...)</elt2>
  <elt3>(...)</elt3>
</elt0>
```

Exemples d'utilisations non valides :

```
<elt0>
  <elt1>(...)</elt1>
  <elt3>(...)</elt3>
</elt0>
```

... car l'élément **elt2** est manquant.

```
<elt0>
  <elt1>(...)</elt1>
  <elt3>(...)</elt3>
  <elt2>(...)</elt2>
</elt0>
```

... car l'ordre des éléments n'est pas respecté.

III-E - Choix d'éléments

Un choix d'élément donne... le choix dans une liste de plusieurs éléments possibles. L'utilisation précise dépend des indicateurs d'occurence. De même que pour la séquence, les éléments-enfants doivent être déclarés dans la **DTD**. Cette liste est composée d'éléments séparés par le caractère | (combinaison de touches **AltGr+6** sur un clavier AZERTY).

```
<!ELEMENT elt0 (elt1 | elt2 | elt3)>
```

Exemple d'utilisation valide :

```
<elt0><elt2>(...)</elt2></elt0>
```

Exemple d'utilisation non valide :

```
<elt0>
  <elt2>(...)</elt2>
  <elt3>(...)</elt3>
</elt0>
```

III-F - Indicateurs d'occurence

III-F-1 - Syntaxe

Lors de la déclaration de séquence ou de choix d'éléments, à chaque élément enfant peut être attribuée une indication d'occurence (**?**, **+** ou *****).

Exemples d'indicateur d'occurences :

```
<!ELEMENT elt0 (elt1, elt2?, elt3+, elt*)>
```

- **elt1** ne comprend aucune indication d'occurence. Il doit donc apparaître une *seule et unique fois* dans l'élément **elt0** ;
- **elt2** a pour indication d'occurence **?**. Cela signifie que l'élément doit apparaître *au maximum* une fois (il peut ne pas apparaître du tout) ;
- **elt3** a pour indication d'occurence **+**. Cela signifie que l'élément doit apparaître *au moins* une fois ;
- **elt4** a pour indication d'occurence *****. Cela signifie que l'élément peut apparaître autant de fois que l'auteur le désire, voire pas du tout.

III-F-2 - Exemples

Les indicateurs d'occurence peuvent être utilisés en conjonction avec les séquences ou les choix d'éléments. Ainsi...

```
<!ELEMENT elt0 (elt1+, elt2*, elt3?)>
```

... permet d'indiquer une séquence composée d'au moins un élément **elt1**, puis d'un nombre indéterminé d'éléments **elt2** (éventuellement nul), enfin au plus un élément **elt3**.

Exemple d'utilisation d'un choix d'éléments avec indicateurs d'occurence par élément :

```
<!ELEMENT choix.elt (elt1* | elt2* | elt3*)>
```

Exemple d'utilisation valide :

```
<elt0>
  <elt2>(...)</elt2>
  <elt2>(...)</elt2>
</elt0>
```

Exemples d'utilisation non valide :

```
<elt0>
  <elt3>(...)</elt3>
  <elt2>(...)</elt2>
</elt0>
```

```
<elt0>
  <elt2>(...)</elt2>
  <elt3>(...)</elt3>
</elt0>
```

Exemple d'utilisation d'un choix d'éléments avec indicateur d'occurence global :

```
<!ELEMENT elt0 (elt1 | elt2 | elt3)*>
```

Exemple d'utilisation valide :

```
<elt0>
  <elt2>(...)</elt2>
  <elt3>(...)</elt3>
  <elt1>(...)</elt1>
</elt0>
```

Dans ce dernier cas, il n'y a pas de contrainte visible sur l'ordre d'apparition des éléments. C'est la déclaration la plus souple possible.

III-G - Élément quelconque

L'élément quelconque est l'élément-« fourre-tout » dans une **DTD**. Il peut contenir tout autre élément défini dans la **DTD**, aussi bien qu'être vide ou contenir du texte. Les éléments-enfants éventuels peuvent apparaître dans n'importe quel ordre, et en quantité non définie. Il est préférable de ne pas utiliser trop souvent ce type de déclaration, car on perd les avantages qu'offre la rédaction d'une **DTD**, qui sont de fixer des contraintes précises sur la structure du document **XML** qui lui est lié. Le mot-clef utilisé pour la déclaration de ce type d'élément est **ANY**.

```
<!ELEMENT elt ANY>
```

III-H - Élément à contenu mixte

Un élément à contenu mixte peut contenir aussi bien du texte que des éléments-enfants. Il se présente comme une liste de choix, avec des indicateurs d'occurence bien choisis. Le texte contenu peut se trouver à n'importe quel endroit dans l'élément, et peut être une section **CDATA**.

Exemple de déclaration :

```
<!ELEMENT citation (#PCDATA | auteur)*>
```

Exemple d'utilisation :

```
<citation>
  <auteur>Shakespeare</auteur>Être ou ne pas être
</citation>
```

Exercice 1. Écriture d'une DTD avec éléments

Énoncé

Correction

IV - Déclarations d'attributs

IV-A - Introduction

Comme on peut trouver dans un document **XML** des éléments possédant des attributs, il est normal que la **DTD** permette de définir des contraintes sur ces derniers. On peut déclarer et attacher à un élément donné chaque attribut séparément, mais il est souvent préférable, afin d'améliorer la lisibilité du code, de les réunir sous la forme d'une liste. Chaque attribut défini dans la liste possède un nom et un type. On peut lui donner une valeur par défaut, ou bien le spécifier obligatoire. Le mot-clef de cette déclaration est **ATTLIST**.

IV-B - Type chaîne de caractères

Il s'agit là du type d'attribut le plus courant. Une chaîne de caractères peut être composée de caractères ainsi que d'**entités analysables**. Le mot-clef utilisé pour la déclaration de chaîne de caractère est **CDATA**.

Exemple de déclaration de **CDATA** (nous reviendrons dans un instant sur la signification du mot-clef **#IMPLIED**) :

```
<!ELEMENT elt (...)>
  <!ATTLIST elt attr CDATA #IMPLIED>
```

Exemples d'utilisations :

```
<elt attr="Chaîne de caractères">(...)</elt>
```

```
<!ENTITY car "caractères">
<elt attr="Chaîne de &car;">(...)</elt>
```

IV-C - Valeurs par défaut

Chaque attribut peut être requis, optionnel ou fixe et avoir une valeur par défaut. Les exemples suivants montrent la déclaration d'un attribut appelé **attr** attaché à un élément nommé **elt**.

1. Déclaration d'un attribut avec une valeur par défaut :

```
<!ELEMENT elt (...)>
  <!ATTLIST elt attr CDATA "valeur">
```

Un tel attribut n'est pas obligatoire. S'il est omis dans le fichier **XML** lors de l'utilisation de l'élément **elt**, il est considéré comme valant valeur. Dans cet exemple, si on écrit **<elt>(...)</elt>**, cela est équivalent à écrire **<elt attr="valeur">(...)>/elt>**.

2. Déclaration d'un attribut requis :

```
<!ELEMENT elt (...)>
  <!ATTLIST elt attr CDATA #REQUIRED>
```

Un tel attribut est obligatoire. Son absence déclenche une erreur du vérificateur syntaxique sur le fichier **XML**.

3. Déclaration d'un attribut optionnel :

```
<!ELEMENT elt (...)>
  <!ATTLIST elt attr CDATA #IMPLIED>
```

4. Déclaration d'un attribut avec une valeur fixe :

```
<!ELEMENT elt (...)>
  <!ATTLIST elt attr CDATA #FIXED "valeur">
```

L'utilité d'un tel attribut peut sembler bizarre à première vue, puisqu'il ne peut prendre qu'une seule valeur. Cette fonctionnalité est cependant utile lors d'une mise à jour d'une **DTD**, pour anticiper la compatibilité avec des versions ultérieures.

IV-D - Type ID

Ce type sert à indiquer que l'attribut en question peut servir d'*identifiant* dans le fichier **XML**. Deux éléments ne pourront pas posséder le même atribut possédant la même valeur.

Exemple de déclaration de type **ID** optionnel :

```
<!ELEMENT elt (...)>
  <!ATTLIST elt attr ID #IMPLIED>
<!ELEMENT elt1 (...)>
  <!ATTLIST elt1 attr ID #IMPLIED>
<!ELEMENT elt2 (...)>
  <!ATTLIST elt2 attr ID #IMPLIED>
```

La déclaration précédente interdit par exemple...

```
<elt1 attr="machin"></elt1>
<elt2 attr="truc"></elt2>
<elt1 attr="machin"></elt1>
```

... ainsi que

```
<elt1 attr="machin"></elt1>
<elt2 attr="machin"></elt2>
<elt1 attr="truc"></elt1>
```

IV-E - Type énuméré

On peut parfois désirer limiter la liste de valeurs possibles pour un attribut. On le définit alors comme étant de type énuméré. Donner une autre valeur dans le fichier **XML** provoque une erreur.

Exemple de déclaration d'une liste de choix d'attributs :

```
<!ELEMENT img EMPTY>
  <!ATTLIST img format (GIF | JPEG | PNG) "PNG">
```

Cet exemple déclare un attribut **format** d'un élément **img**. La valeur de cet attribut peut être **PNG**, **GIF** ou **JPEG**. Si aucune valeur n'est affectée à cet attribut, c'est la valeur par défaut qui le sera, ici **PNG**. On notera l'absence de guillemets dans la liste des valeurs possibles. En ajouter est une erreur courante dans la rédaction d'une **DTD**.

IV-F - Utilisation de liste pour les attributs

On utilise le fait qu'il est possible de « factoriser » le nom de l'élément. Par exemple...

```
<!ELEMENT elt (...)>
```

```
<!ATTLIST elt
  attr1 CDATA #IMPLIED
  attr2 CDATA #REQUIRED
>
```

Exercice 1. Écriture d'une DTD avec attributs

Énoncé

Correction

V - Déclarations d'entités

V-A - Introduction

Les déclarations d'entités permettent de disposer de l'équivalent de raccourcis clavier et de caractères a *priori* non accessibles dans le jeu de caractères sélectionné.

V-B - Les entités paramétriques

Elles servent à définir des symboles qui seront utilisés ailleurs dans la **DTD**. Ce sont en quelque sorte des raccourcis d'écriture : partout où une entité est mentionnée, elle peut être remplacée par la chaîne de caractères qui lui est associée. Ce mécanisme s'apparente à un mécanisme de « macro ».

Les entités paramétriques ne peuvent pas être utilisées en-dehors d'une **DTD**.

Exemple tiré de la **spécification du langage HTML** :

```
<!ENTITY % heading "H1|H2|H3|H4|H5|H6">
```

L'exemple précédent a pour effet d'indiquer au système que toute occurence de **%heading;** doit être remplacée par **H1|H2|H3|H4|H5|H6** dans la **DTD**.

Ce mécanisme peut également servir à utiliser un nom relativement compréhensible à la place d'une séquence de caractères peu évocatrice. La définition d'une entité peut également faire référence à d'autres entités ; la substitution est alors effectuée de proche en proche.

V-C - Les entités de caractères

Elle servent à donner un nom facilement lisible à des caractères qui ne sont pas représentables dans l'alphabet utilisé, ou qui ne sont pas disponibles au clavier.

Exemples tirés de la **DTD** du langage **HTML 4.01** :

```
<!ENTITY nbsp " ">
<!ENTITY eacute "&#233;">
```

Les entités de caractères définies dans une **DTD** peuvent être utilisées dans un document **XML** référençant cette **DTD** à l'aide de la notation **&NomEntité;** (par exemple **&ecaute;** pour reprendre une des entités précédentes).

V-D - Les entités internes

Ce sont des symboles pouvant être définis dans une **DTD** et utilisés dans un document **XML** comme raccourcis d'écriture. La définition complète du symbole est entièrement incluse dans la **DTD**. Exemple :

```
<!ENTITY ADN "Acide désoxyribonucléique">
```

Dans le fichier **XML**, l'appel à **&ADN;** sera aumatiquement remplacé, lors de l'affichage ou du traitement, par la chaîne de caractères "*Acide désoxyribonucléique*".

V-E - Les entités externes

Il s'agit...

... soit de symboles pouvant être définis dans un autre fichier, mais pouvant être utilisés dans un document XML ou la DTD elle-même. Par exemple :

```
<!ENTITY Inclusion SYSTEM "toto.xml">
<!ENTITY % Inclusion SYSTEM "toto.inc">
```

Dans le fichier **XML**, le contenu du fichier **toto.xml** sera inséré à l'appel de l'entité **&Inclusion;**, et dans la **DTD**, le contenu du fichier **toto.inc** sera inséré à l'appel de l'entité **&Inclusion;**

... soit de symboles pouvant être définis dans une autre **DTD** et utilisés dans la **DTD** courante :

```
<!ENTITY % HTMLSpecial PUBLIC "-//W3C//ENTITIES Special for XHTML//EN" "http://www.w3.org/TR/xhtml1/
DTD/xhtml-special.ent">
```

Le contenu de cette **DTD** (qui peut être de type **SPECIAL** ou **PUBLIC**) est importé dans la **DTD** courante par l'appel de **%HTMLSpecial;**.

Exercice 1. Déclarations d'entités

Énoncé

Correction (Fichier XML)

Correction (DTD)